Trabajo 3

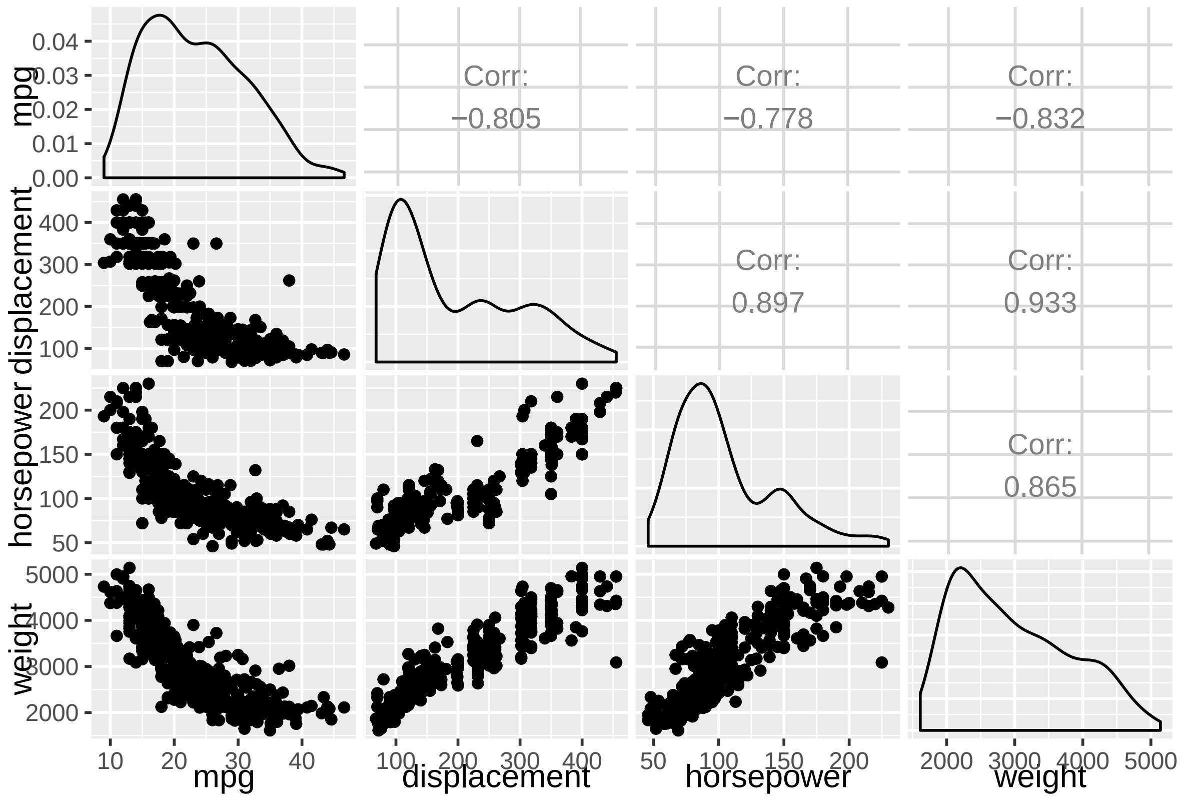
Antonio Álvarez Caballero

27 de mayo de 2016

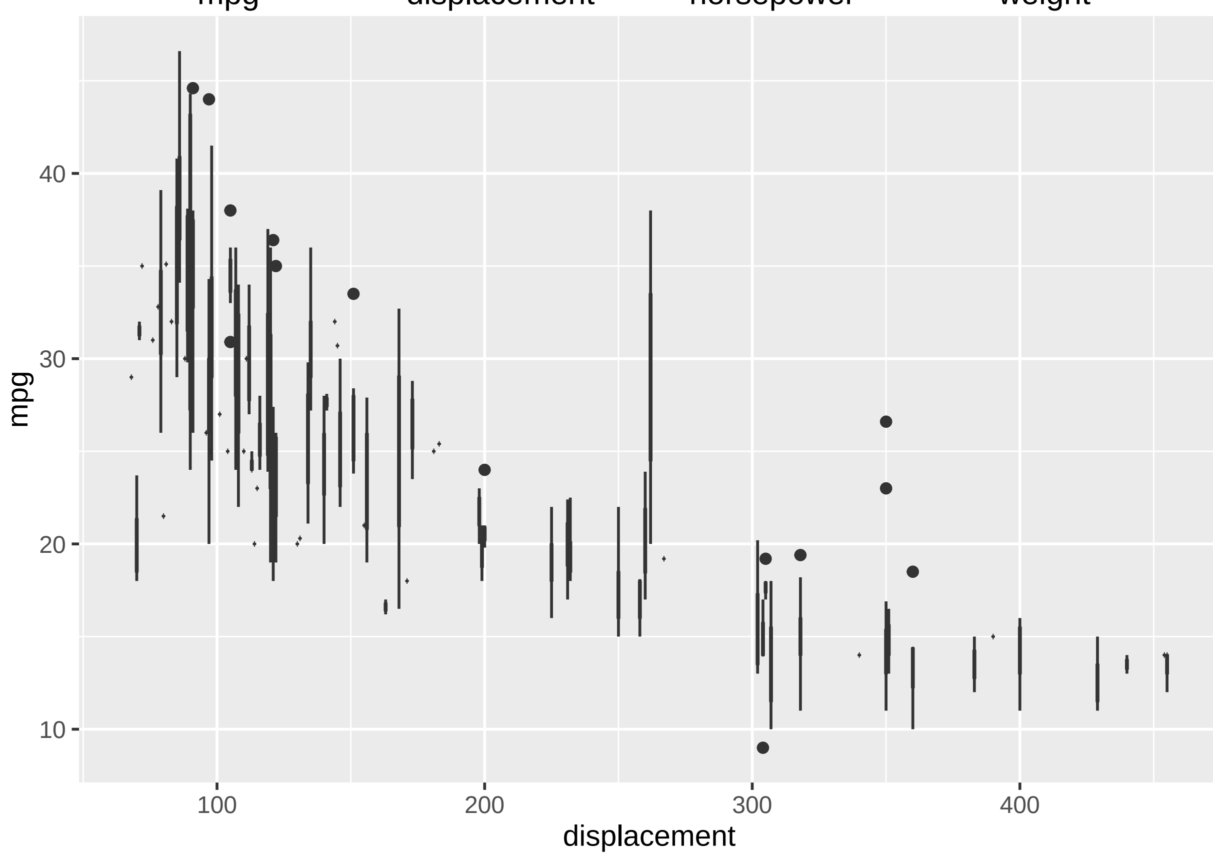
Ejercicio 1

Apartado a)

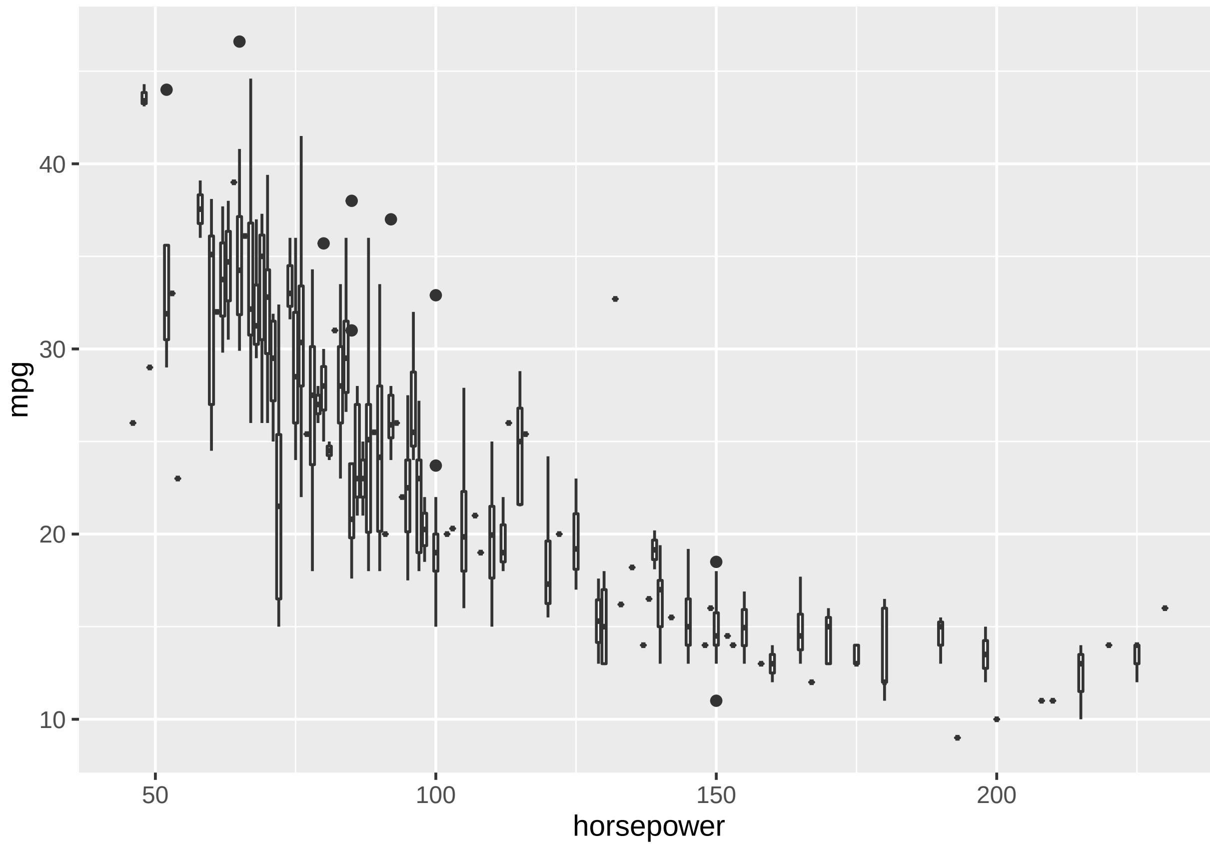
Parece ser que las variables de las que más depende *mpg* son *displacement*, *horsepower* y *weight*. Veámoslas con más detalle.



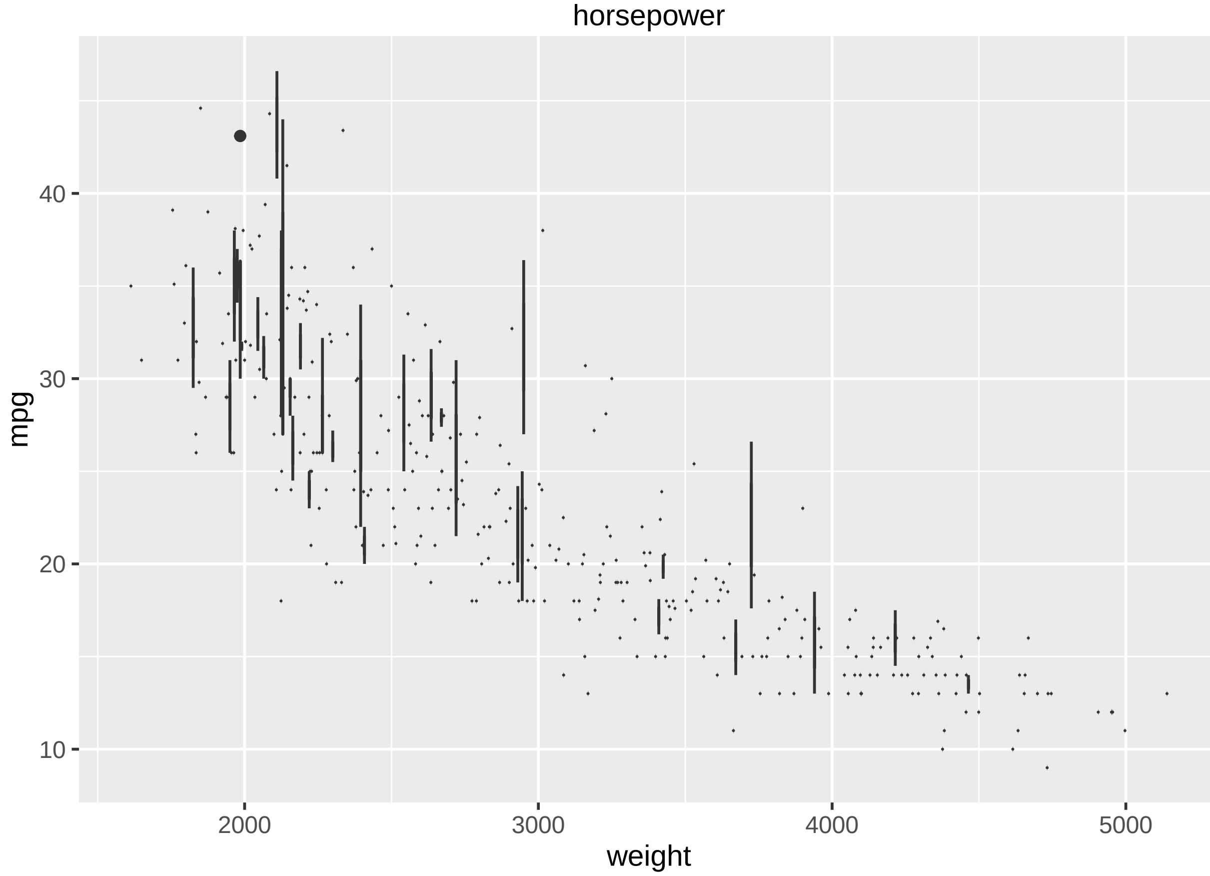
Corr:
−0.805
Corr:
−0.778
Corr:
−0.832
Corr:
0.897
Corr:
0.933
Corr:
0.865
mpg
displacement
horsepower
weight



mpg
displacement



mpg
40
30
20
10
50
100
150
200
horsepower



mpg
40
30
20
10
2000
3000
4000
5000
weight

Apartado b)

Seleccionamos las variables que hemos decidido para predecir.

```
Auto.selected <- Auto[,c("displacement","horsepower","weight")]
```

Apartado c)

Como nuestro conjunto de datos es grande (392 instancias), podemos realizar un muestreo aleatorio. Así tampoco falseamos las muestras, cosa que podría pasarnos si realizamos un muestreo estratificado.

```
index <- sample(nrow(Auto), size = 0.8*nrow(Auto) )

Auto.train <- Auto.selected[index,]
Auto.test <- Auto.selected[-index,]
```

Apartado d)

Vamos a crear una nueva variable, *mpg01*, la cual tendrá 1 si el valor de *mpg* está por encima de la mediana y -1 en otro caso.

```
mpg01 <- ifelse(Auto$mpg >= median(Auto$mpg), 1, -1)
Auto.selected$mpg01 <- mpg01
Auto.train$mpg01 <- mpg01[index]
Auto.test$mpg01 <- mpg01[-index]
```

Apartado d1)

Vamos a ajustar un modelo de regresión logística para predecir *mpg01*.

```
model.LogReg <- glm(mpg01 ~ ., data = Auto.train)
prediction.LogReg <- predict(model.LogReg, newdata = Auto.test)
prediction.LogReg
```

```
##            5           10           12           26           37
## -0.5577879948 -0.9848113013 -0.7381853840 -1.2661383548 -0.4088781960
##           38           44           45           48           49
## -0.3102376693 -1.5851145954 -1.8021749451 -0.3698870666 -0.3143571077
##           58           60           64           71           73
##  0.6813820684  0.6578193980 -1.3643625422 -1.3515764370 -0.7988630221
##           79           80           87           90           92
##  0.2320710500  0.7332608189 -0.6712885470 -0.7812764656 -1.4672430636
##           96           99          100          103          107
## -1.7713307527 -0.3675675307 -0.1113374650  0.8158418272 -1.2436978748
##          108          116          122          123          124
## -0.0208755646 -1.0817838183 -0.5620803222  0.4660510009  0.2854512537
##          125          132          137          146          152
## -0.7594947537  1.0165074646 -0.9590677072  0.8684498897  0.8979148668
##          154          155          165          167          180
## -0.4611125932 -0.5207877128 -0.1395115340 -0.4205311435  0.2733906175
##          187          198          201          202          206
##  0.7401456017  0.9027172691 -0.5894345127 -0.5575570755  0.7631664490
##          219          221          224          230          235
```

```
##  0.9788494754  0.9156139476 -1.0031882868 -1.2572677485  0.2642244559
##           239           243           246           249           250
##  0.8243123170  0.5008440395  0.9449730235  0.9558263490 -0.4302610625
##           258           271           277           282           287
## -0.2878345939  0.4702994144  0.3991806014 -0.0594452542 -0.7429466349
##           300           318           333           341           347
## -0.0004592205  0.7508786402  0.9413114452  0.3254059328  0.7970937071
##           350           353           358           359           360
##  0.8668101131  0.6063575291  0.4763322051  0.4018749008 -0.0031094922
##           368           372           376           378           382
##  0.4792876455  0.4358827622  0.8436147540  0.7251123179  0.6609843029
##           385           388           393           397
##  0.8587284859  0.3457018720  0.2692433328  0.3743542127
```

```
error.test <- "¿Esto cómo va?"
```

El error de test de este modelo es `¿Esto cómo va?`.

Apartado d2)

Ahora vamos a ajusart un modelo k-NN.

Apartado d3)

Veamos las curvas ROC de ambos modelos.

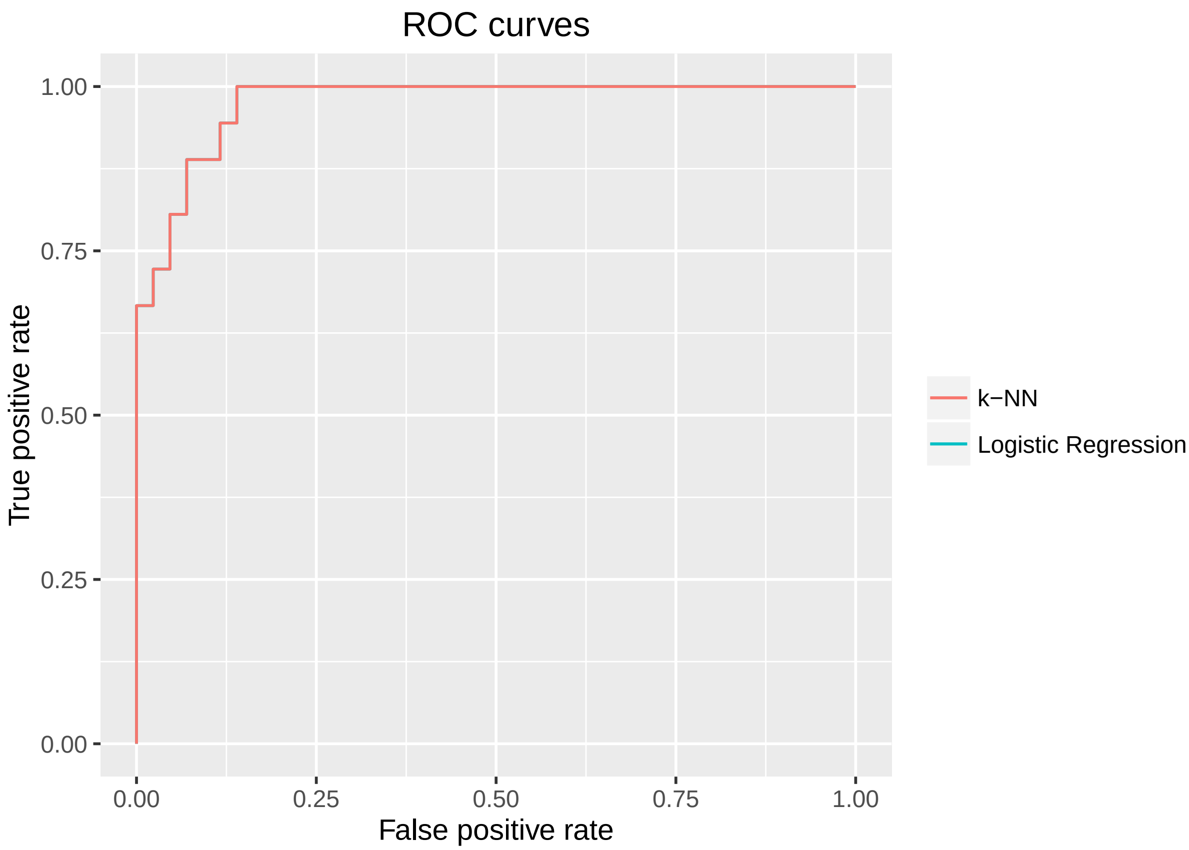
```
roc.prediction.regLog <- prediction(prediction.LogReg, Auto.test$mpg01)
roc.performance.regLog <- performance(roc.prediction.regLog, measure = "tpr", x.measure = "fpr")

### Meter del knn

####
roc.performance.knn <- roc.performance.regLog

roc.data <- data.frame(x  = roc.performance.regLog@x.values[[1]],
                       y1 = roc.performance.regLog@y.values[[1]],
                       y2 = roc.performance.knn@y.values[[1]])

ggplot(roc.data, aes(x)) +
  geom_line(aes(y = y1, colour = "Logistic Regression")) +
  geom_line(aes(y = y2, colour = "k-NN")) +
  theme(legend.title = element_blank()) +
  labs(title= "ROC curves", x = "False positive rate", y = "True positive rate")
```




```
auc.regLog  <- auc(roc.data$x,roc.data$y1, type = 'spline')
auc.knn     <- auc(roc.data$x,roc.data$y2, type = 'spline')
```

El área bajo la curva de la ROC de regresión logística es `0.9717848` y la del k-NN es `0.9717848`. Luego `k-NN` es el modelo que mejor *performance* tiene.

Apartado e) (BONUS)

Para estudiar el error con validación cruzada hacemos uso de `cv.glm`

```
model.full.LogReg <- glm(mpg01 ~ ., data = Auto.selected)
cv.LogReg <- cv.glm(data = Auto.selected, glmfit = model.full.LogReg, K = 5)
cv.LogReg
```

```
## $call
## cv.glm(data = Auto.selected, glmfit = model.full.LogReg, K = 5)
##
## $K
## [1] 5
##
## $delta
## [1] 0.4141693 0.4133581
##
## $seed
##   [1]         403         314 -1979582993  -233419227  -359642404
##   [6] -1129013070  1479783341  1637607271 -1484778354  1525453160
##  [11] -1480715925   478831369   918230424   -67001146  2009030177
```

```
##  [16]   -983253325   -915151838   134831636  1853149847  2014022797
##  [21] -1461354076 -1052447862   533907445 -2007415217   428671318
##  [26] -1232253952  1186415939  1833101537  2142650960  -215589202
##  [31] -1298494887 -1928719765 -1248938022 -1983268708   576846495
##  [36] -1252419339 -1531936372  2026876002   179616829 -1099002185
##  [41] -1269109410   984118424   -80306181  -738490983  1050318376
##  [46] -1628254506  1274407761   534785539  1573573010 -1449554140
##  [51] -1116347929  1601441341   657348788  -424939750  -950909851
##  [56]   703073535  1366687110 -1494420720  1808336819 -1360384943
##  [61]  1470140416  2100117726  1881914153  -308525349   184448490
##  [66] -2088854132 -1807576369  1851477637 -1352372932 -2129154030
##  [71]   -942833907   -553763769  1298681390  -876046456   259744395
##  [76] -1419783063  2094192696    48845030 -2137638783  1880498195
##  [81]   297035714  1349468276  -587565449  1977706349  -289908732
##  [86]  1582322474  1577150613  1144370991  1145855542  1992804960
##  [91]   -819668829   -420660287   -24676432   740074190  1945944121
##  [96] -1502464437   -709641222  -837122308 -1070085313   444056597
## [101]   -158900436   -493177854  2081683165 -1272003113  1391341758
## [106]  1292190456    -81377061  1556396857  -900804984   990722614
## [111]   -128118159 -1529378653   412697074  1627377988   288606727
## [116]  1346156509 -1851592492    10285050   856516421 -1287781473
## [121]   251382054   154278320  -754666925  -190612111  1754254496
## [126]   -283292610 -1314880695 -1628859909  1050015178  1688124204
## [131]  1957770031   791341541 -1004260196  1615511794 -1650108051
## [136] -1635281241   976296142   702756392  -895009237  1171838025
## [141] -1257069224  1292516358  1798421985  1277888371  -924449822
## [146]  1769188180  1300571735  1109171789 -1576889884   521038794
## [151] -2035249611 -2038344561   288210326  -505487040   290208771
## [156] -1348665823  1543244048   101164782  -307381607   989737771
## [161]   -984785382  1788474588     -1537825 -1956436299  1435000908
## [166] -1139254110   -589337603  -946521865 -1081860578    72659160
## [171]    87446587   -645218855   189000552  -346093930  -723001327
## [176]   178241987   609218002  1721910372 -1537182809   867507453
## [181] -1199064844   -673198118  1228841765   901354047 -2008172090
## [186] -1728639408  1177600627  1350809233 -1760637248 -1057040226
## [191] -1876937623   -641458021  -400350806   354990540  1045632143
## [196]  1797352645 -1247150980   851439826  -667152307   928737543
## [201]   -416965394   -825379384    44939979  1676141353 -2028403848
## [206]   -914171738   -744119615  1181586259  1771732994  -508678860
## [211] -1991055689  1992841901   -52782140  -546208790  -576983979
## [216] -1356066321  1070184950   549822240  2115334371   483894657
## [221]  1337770992  1505397390  -218997255   194502539  1789320122
## [226]   680149692   885168639  -517781675  2078474348  1584476968
## [231] -1348985286  1186873808 -1076558484 -1857610828  2029227026
## [236]   789535200  1972714444  1645041888   453906466 -1649947352
## [241]   -267031028   -351581380  -681925646   805433072  1070494276
## [246] -2089619880  1087959866  1333633520 -2132623684  -547255020
## [251]   -383625534   -545925024 -1002281092 -1087837232  1318800674
## [256] -1899407704   -328356980   200948924  -858626254    19036736
## [261]   -309075788 -1315706296 -1853715014    90972688  1658779372
## [266]   -810502828   -349440750    87177632  1691293388   421670496
## [271]   -784696382   -368284888  -513082452 -1880531140   236420466
## [276] -2122715344 -2106362588   933323576  1487576794  -715510896
## [281] -1972118788  1201611668   794894402  -676978432  -178045892
```

```
## [286] -1273260720 -1672687454  -919878328  1643268876  -685036836
## [291] -1698664110  -389699456  -251130796   226026472  1345076474
## [296]  2124087120  -829958356  1843076468  -839278702 -1876558752
## [301]  -448961140  -600648864  -284297758 -1495694424   176990412
## [306]   116741116   255754674 -1914483216   322297796  1823410520
## [311]   108944762   289097648   998194620 -1725500332   957863874
## [316] -1994110944   375423164  2039248464 -2080253982  1280710760
## [321]  1624168524 -2051881732 -2136201230  1959304448 -1488050828
## [326]   295936008  1658364858   468125136 -1802469396 -1141561964
## [331]  -140238382  -868787872   152312652  1594360096   818177922
## [336]  2011643624  1025976556    81102908  -796291214   -65851792
## [341]   645778468 -1489756360   224251034  1026757328  1196101820
## [346]  -717869420  -898847102  -789618496   825460540  1189679248
## [351]  -204206558  -816649208 -1764815604  1400343708  -754740718
## [356]  -316820608   150877716  -804426200  1759780154   623177168
## [361]   367413356  1109034292 -2090030446 -2137305248   524726092
## [366]  1466329056   977468322   516447144   597342604 -1808401220
## [371]   503873906   196077808  1522456644  -907975336   242076218
## [376]  2005634928  1492356156   749208340 -1280883262  -947073312
## [381] -1940640900 -2032110896  -158438238 -1792655320  1116600588
## [386]  2099923900   183647538  2016567744  2141174324   269930440
## [391] -1925104454  -231706224 -1330003476   711079508  -660499310
## [396]  1893237536   298451404  -870631968 -1285511486  -910669784
## [401]  -434481492   941711548   899341810 -1460646608   420144292
## [406]  1469758136  1571805530  1823459728 -1189943428  -573970156
## [411]    85008194    48279296 -1814445124 -1109159088   638189346
## [416]   841508168 -1554510324  1429846108  -820768558  1985531392
## [421]   664359252 -1366991384  1490402298   228383952  2108291372
## [426]  2081507060  1026055570   313294944  -349891316  -678184224
## [431]  1549499746   575546024  -898192820   231929212   568547634
## [436]  -949105552  -170816700  1547805144   593597562   578856496
## [441] -1006615876  1185706836 -1630916542 -1603614304  1880897212
## [446]  -761758256   947731810 -1980332056   979972044  2128298748
## [451]   441824754 -1321333632  1648695540 -1541219704  -693081286
## [456] -1000306864  2034564486   114555315   772576533   639293298
## [461] -1289194560 -1039944663   450027675 -1293005460   675591546
## [466]  -371673425 -1544856135  1042677710  -105525476  2100788877
## [471]   208379639 -1640640000 -2140840866  -103178453   194473005
## [476]  1941052570 -1424568232  1085568961 -1857398669  1059541396
## [481]  -661381086 -1544591561  -123213311  1200599814  1395695428
## [486]  1565143669  1550541791   938395448 -1530986218  1623227683
## [491]  1301524197 -1072228606  -206586512   843324697   774591755
## [496] -2025056772  1960046122   429291103  -775551511  1863923454
## [501] -1623693588 -2041244099 -1060026873  1453689328  1196136846
## [506] -1807863877 -1365722723  1564342090  1718405992 -1974743727
## [511] -1816994237  2036482884  -602546190  1385799111 -1454709871
## [516]   -77276394  -651890828   379194885 -1079560625   853021256
## [521]   576685030  -346936045  2130682485    58986578  1949852320
## [526]  1552813961  -216118341  1261865996 -1901774438   864721167
## [531]  -129112231  -111015442   500601980 -2105642771  -413976681
## [536]  -707964832   716739134  -263752949  2135042445 -1302015302
## [541]  -501368584  -569227615  -170697133 -1155800716   184047362
## [546]  1711849367  1030992225 -1811891802 -1749517404   604907093
## [551]  1256400639    64416664   821202102 -1043044029  1914126469
```

```
## [556]  1138301218  -480384496   261583289 -2077769109  -131457700
## [561]  1860201610 -1684560769  1557220809  -172210594   821536012
## [566]   559456477 -1629565465 -2005821936 -1627550930 -2091659301
## [571]   103313597 -2061895126 -1928354104  1228850161 -2057394845
## [576] -1002652316   678641810   494441831  -551899343   927401078
## [581]   290463124   119515557  -487734609 -1148222616 -1109704890
## [586] -1715247373    59287637  -236022734  1228960256  2013102825
## [591]  -369141413 -1764764756   216608698 -1014380177  1928643321
## [596]  -483202162 -1072851364   -25508659   820186167  -884070720
## [601] -1865580130 -1036895637  1916789229 -1471369126  1785896728
## [606]   430466177  -694304205  1081892948  -758340510   705047287
## [611] -1099778751 -1553820730  1291350276 -1627991883   147775391
## [616]  -386314248  2096366038   276633699 -1349178331  1767001026
## [621]  1351073968  1329425625   232434251 -1467864900  -115126550
## [626]   795405546
```